07

Para saber mais: Refactoring

No processo de desenvolvimento de software passamos grande parte do tempo lendo código. Muitas vezes esse tempo é maior do que o tempo gasto escrevendo novas linhas de código. Isso acontece pois, para qualquer alteração, é preciso procurar quais classes ou arquivos serão atingidos, entender qual é o funcionamento atual e como adicionar o novo comportamento sem mudar o antigo.

Uma consequência direta disso é que quanto maior a dificuldade em ler e entender uma parte do código, maior é o tempo gasto para fazer uma alteração nela. Mesmo se a pessoa que fez o código for a mesma a fazer a alteração, ela precisa de algum tempo lendo o código para lembrar o que foi feito.

Para evitar esse problema é preciso manter a qualidade e a clareza do código sempre altas, diminuindo assim o custo para fazer alterações no código. É preciso melhorar o código já pronto sem mudar o seu comportamento, o que chamamos de **Refactoring(Refatoração)**.

O objetivo da refatoração é melhorar algum aspecto do código com o cuidado de não mudar o comportamento atual do sistema. Exemplos de aspectos que podem ser melhorados: legibilidade, clareza, baixo acoplamento, separação de responsabilidades, etc.

Refatorar sem ter uma garantia concreta de que o comportamento do código não vai mudar é um pouco imprudente, pois não adianta nada melhorar o código se algo que já existe é quebrado. Portanto é indispensável que existam testes automatizados para o código que está sendo refatorado.

Conheça mais detalhes sobre o processo de refactoring acessando os seguintes sites: Refactoring (https://refactoring.com) e Refactoring.guru (https://refactoring.guru/pt-br/refactoring).