É muito comum vermos desenvolvedores que estão começando com a prática de TDD abusar dos baby steps. Um exemplo disso é o seguinte:

Imagine que precisemos criar um método simples, que realiza uma soma. Poderíamos começar testando se 2 + 2 é igual a 4. Assim que o teste falha, o programador escreve o código mais simples que faz o teste passar. Nesse nosso caso, a implementação abaixo faz o teste passar:

```
public int soma(int a, int b) {
   return 4;
}
```

Em seguida, para continuar a trabalhar na implementação, ele escreve o teste 3+3 é igual a 6. Com o teste falhando, o programador faz a seguinte implementação:

```
public int soma(int a, int b) {
   if(a == 3) return 6;
   return 4;
}
```

E continua de maneira análoga. Veja que o programador está sempre tentando escrever o código mais simples possível.

O que você acha do exemplo acima? Quais são as vantagens e desvantagens?

Opinião do instrutor

A ideia dos baby steps serve justamente para que o programador consiga dar passos pequenos o suficiente para que ele não se perca, mantenha foco, e produza código de qualidade.

Infelizmente, muitos programadores levam essa ideia ao extremo, e acabam por escrever códigos como os de cima (que na verdade não é simples!). Ao tomar passos realmente tão pequenos O TEMPO TODO, o programador acaba por diminuir sua produtividade.

O próprio Kent Beck, autor do livro mais famoso sobre TDD, comenta que você deve dar baby steps somente quando precisa, quando se sente inseguro em relação ao código que está produzindo.

Sua experiência te dirá a hora de dar um passo maior ou mais devagar.