Notas sobre o capitulo 2 de Clean Code – by Lucas Trevizan

Introdução

Damos nomes para tudo, é assim que reconhecemos as coisas, esse capítulo abordará dicas para se escolher bons nomes.

Use nomes que revelem seu propósito

Parece óbvio, como de fato é. O nome de uma variável, classe e função tem que descrever com exatidão o que ela faz, se você achar um nome melhor depois de um tempo, então mude, se precisar de um comentário para explicar, então está ruim.

lixo: *int d; //tempo decorrido em dias*

bom: *int tempoDecorridoEmDias*;

O livro tem um excelente método de exemplo para explicar a importância de nomes, mas basicamente começamos com um método confuso de ler onde os nomes não te dizem o contexto. Pensando assim, vale ressaltar a importância de um método ao nomeá-lo respondendo às perguntas que o código original deixa em aberto. A cada nome semântico dado, o método se torna mais legível.

Evite informações erradas

Abreviar nomes não é o melhor caminho a se seguir, o que parece óbvio para você em uma abreviação pode ter um significado dúbio para outo. Se eu falar "hp" por exemplo, você pensa na marca de impressoras, no Harry Potter, na hipotenusa... acho que já deu para entender. Use nomes completos, cuidado ao usar o "list" nos nomes, sabemos que list é algo específico para programadores e cuidado com nomes parecidos. Alguns caracteres podem gerar confusão, um "l" minúsculo se assemelha ao 1 e "o" também minúsculo se assemelha a um 0.

Faça distinções significativas

Se os nomes precisam ser diferentes, também devem ter significados distintos. Resumidamente falando, não use palavras genéricas tipo Data, Info etc.. para complementar uma classe ou método, como você entenderia a diferença de cada método abaixo:

getInfoConta(); *getConta()*, *getDataConta()*.

Então pensando de novo no conceito de ser um autor de um código, escreva as coisas com o pensamento de que o leitor saiba o que você esta dizendo, e portanto, entenda a diferença.

Use nomes pronunciáveis

Quanto mais abreviada sua variável for, mais um idiota você vai parecer ao pronunciar ela durante uma discussão. Então pense em nomes inteligentes para que surjam conversar inteligentes ao discutir situações que envolvam essa variável, eu programo em português então não me faltam palavras.

Use nomes passíveis de busca

É muito mais fácil você procurar no código um nome que faça sentido do que procurar um número ou uma letra comum como "e". Bob fala que variáveis com uma única letra devem ser usadas apenas em escopo local dentro de métodos pequenos, eu acho que não devem ser usadas nunca.

A notação Húngara

Notação Húngara era eficaz quando as linguagens não possuíam um nível maior, existia limite de caracteres para nomear variável e outros tipos de limitações. Com o avanço da linguagem e das próprias IDE's a notação passou a ser um indicativo de um código "velho".

Interfaces e implementações

Não enfeite interfaces com nomes que dão informações excessivas ou distrativas, como por exemplo *IShapeFactory*.

Evite o mapeamento mental

Variáveis de letra única possuem certos padrões que um programador exibido gosta de mostrar, então ele faz um método pequeno com "a", "b" ,"i", "j" todas guardando pequenos valores que você constantemente precisa lembrar e mapear durante a leitura. Eu particularmente detesto essa abordagem e esse trecho em específico relata o que está se tornando redundante na leitura: "ESCREVA CÓDIGOS QUE OS OUTROS POSSAM ENTENDER!!!!".

Nome de classes

Nomes de classe e objetos precisam ser substantivos (*que evidencia a substância, a essência)*. Evite palavras como Info, Dados etc.

Nome de métodos

Métodos devem ser verbos que descrevam com exatidão o que aquele método faz.

Não de uma de espertinho

As vezes pode parecer divertido colocar um nome de método como uma piada interna, NÃO FAÇA ISSO! Mantenha sempre a clareza no nome, expresse o que você quer de fato expressar, não de margem ao erro.

Selecione uma palavra por conceito

Escolher uma palavra por cada conceito abstrato ajuda a manter a coesão do código, ter métodos com palavras semelhantes como *"pegar"* " *"recuperar"* e *"manter"* gera confusão ao ter que decidir o que de fato você precisa utilizar. Da mesma forma acontece com as classes, então mantenha sempre um "dicionário" consistente.

Não faça trocadilhos

Não use a mesma palavra para dois propósitos, um termo com dois propósitos diferentes é um trocadilho. Tudo bem usar o mesmo nome de método em várias classes, por exemplo um "add" quando ele representa a mesma semântica. Agora se um método não faz semanticamente a mesma coisa que o método "add", então é melhor usar outro nome.

Use nomes a partir do domínio da solução

Tudo bem usar termos padrões de informática. Selecionar nomes técnicos para atividades técnicas, é o método mais adequado.

Adicione um contexto significativo

Poucos nomes são significativos por si só, por isso usamos o contexto para deixar mais legível. No livro temos um exemplo grande de código mostrando como melhorar um método que à primeira vista não é tão claro.

Não adicione contextos desnecessários

Não adicione mais contexto a um nome do que é necessário, isso pode dificultar um pouco o trabalho da sua ide, nomes curtos geralmente são melhores desde que sejam claros.

Conclusão

Escolher bons nomes está diretamente ligado a uma boa habilidade de descrição, e isso é mais uma questão de prática do que técnica. Se precisar renomear variáveis por saber que isso melhorará a legibilidade do código, faça.