As funções (ou métodos) são a primeira linha de organização em qualquer programa. Escreve-las bem é de suma importância. Veremos como podemos fazer isso.

Pequenas

Eu posso afirmar que funções pequenas são melhores para o leitor conseguir entender o programa. Já fiz funções grandes e muito grandes, de mais de 100 linhas, e afirmo que são desgastantes de se ler e dar manutenção. Podemos reduzir nosso código para novos níveis de abstração, para que ele fique mais legível, talvez criar uma outra função especifica para algo, e chamar ela na função atual.

Faça apenas uma coisa

O conselho a seguir tem aparecido de uma forma ou de outra por 30 anos ou mais. ***As funções devem fazer uma coisa. Fazê-la bem. Devem fazer apenas ela.***

O maior problema aqui seria em definir “uma coisa”, o que seria “uma coisa”? Se formos levar ao pé da letra, certamente chegaremos a lugar nenhum. Até porque não escrevemos o código de imediato pensando nesses princípios, então vamos pensar que já o escrevemos. Nesse caso será mais fácil ver se o código faz mais de uma coisa ou não.

Na minha visão e interpretação, “fazer uma coisa”, pode se referir ao objetivo da função. Por exemplo, em um método “MovePlayer” não irei colocar coisas que alterem a animação do player, pois seria mais de uma coisa. Eu posso dentro do mesmo objeto ter uma função “AnimationPlayer”, que eu posso chamar dentro do objeto.

Uma função com o objetivo de mover o player, e outra para controlar a animação, tenho os dois separados, e posso chamar o “AnimationPlayer” quando precisar, pois, ele não estará ligado com o “MovePlayer”.

Use nomes descritivos

Já falamos da importância dos nomes, e aqui vamos reforçar isso. Porém também devo dizer para não gastar tempo com isso, pois você poderá tentar vários nomes, e mesmo assim reescrevê-los. Muitas vezes quando trabalhamos mais no código conseguimos ver onde erramos, e então criar nomes melhores e mais objetivos, com as IDEs modernas, trocar nomes se tornou fácil.

Crie nomes extensos sem preocupações, pois eles são melhores que curtos e enigmático. Nem sempre uma função será clara como “SetScore” por exemplo, então usar nomes extensos, mas claros e objetivos será uma boa saída para deixar o código bom.

Parâmetros de funções

Muitas funções criadas exigem parâmetros. Seja ela feita por nós ou de bibliotecas e frameworks. Os construtores ajudam a lidar com quantidades diferentes de Parâmetros nas funções, por vezes precisamos de um ou vários parâmetros para a classe.

Mas estamos falando de funções, e aqui pode não ser tão necessário quanto nem elegante ter diversos parâmetros para usar a função desejada. É provável que se uma função precisar de mais de uma função, ele(s) pode(m) ser colocados em uma classe própria. Assim reduzindo o número de parâmetros através da criação de um objeto.

Separação comando-consulta

As funções devem fazer ou responder algo, mas não ambos. Sua função ou altera o estado de um objeto ou retorna informações sobre ele. Efetuar as duas tarefas costuma gerar confusão. Uma simples situação seria o placar de um jogo “Score”, criaríamos uma função “GetScore” para leitura, e um “SetScore” para alterar o valor.

Evitar repetições

Imagine uma situação em que temos diversas funções que fazem a mesma coisa, mas com nomes diferentes sendo usadas no mesmo código. Seria desagradável para qualquer um que for ler isso. A duplicação pode ser a raiz de todo mal do software. Muitos princípios e práticas tem sido criados com a finalidade de controlá-la ou eliminá-la.

Como escrever funções?

Criar um software é como qualquer outro tipo de escrita. Ao escrever um artigo, você primeiro coloca seus pensamentos no papel e depois os organiza de modo que fiquem fáceis de ler. O primeiro rascunho pode ficar desorganizado, então você reestrutura e refina até que ele fique como você deseja.

Quando escrevo funções, elas começam longas e complexas; há muitas endentações e loops aninhados; possuem longas listas de parâmetros; os nomes são arbitrários; e há duplicação de código.

Sedo assim, eu organizo e refino o código, dividindo funções, troco os nomes, elimino a duplicação, reduzo os métodos e reorganizo, por vezes desmonto classes inteiras.

No final, minhas funções seguem as regras que citei acima, mas não as aplico desde o começo, não acredito que seja possível.