Sumário

Prefácio **11**

Capítulo 1 ▪ Códigos devem ser fáceis de entender **15**

O que torna um código "melhor"? 16
Teorema fundamental da legibilidade 17
Menor é sempre melhor? 17
Por acaso o tempo-para-entender entra em conflito com outros objetivos? 18
A parte difícil 18

Parte I ▪ Melhorias superficiais **19**

Capítulo 2 ▪ Criação de nomes informativos **20**

Escolha palavras específicas 21
Evite nomes genéricos como tmp e retval 23
Prefira nomes concretos a nomes abstratos 26
Inclusão de informações extras a um nome 29
Qual deve ser o comprimento de um nome? 32
Utilize a formatação dos nomes para transmitir significado 34
Sumário 36

Capítulo 3 ▪ Nomes que não podem ser mal interpretados **37**

Exemplo: Filter() 38
Exemplo: Clip(text, length) 38
Prefira min e max para limites (inclusivos) 39
Prefira first e last para intervalos inclusivos 40
Prefira begin e end para intervalos do tipo inclusivo/exclusivo 40
Nomenclatura de booleanos 41
Atendendo às expectativas de seus usuários 42
Exemplo: avaliação de vários candidatos a nomes 44
Sumário 46

Capítulo 4 ▪ Estética 47

Por que a estética é importante? 48
Reorganize quebras de linhas para que sejam consistentes e compactas 49
Utilize métodos para eliminar irregularidades 52
Utilize o alinhamento em colunas quando adequado 53
Escolha um ordenamento significativo e utilize-o de modo consistente 54
Organize declarações em blocos 55
Divida seus códigos em "parágrafos" 56
Estilo pessoal versus consistência 58
Sumário 59

Capítulo 5 ▪ Como saber o que comentar 60

O que NÃO devemos comentar 61
Registre seus pensamentos 64
Coloque-se na posição do leitor 67
Considerações finais – como superar seu bloqueio de escritor 72
Sumário 73

Capítulo 6 ▪ Crie comentários precisos e compactos 74

Mantenha seus comentários compactos 75
Evite pronomes ambíguos 75
Melhore referências imprecisas 76
Descreva o comportamento das funções de modo preciso 76
Utilize exemplos de entrada/saída para ilustrar situações confusas 77
Declare a intenção de seu código 78
Comentários de "parâmetros de função nomeados" 79
Utilize palavras informativas 80
Sumário 81

Parte II ▪ Simplificação de loops e lógica 82

Capítulo 7 ▪ Como facilitar a leitura do fluxo de controle 83

Ordem dos argumentos em condicionais 84
Ordem de blocos if/else 85
Expressão condicional ?: (também conhecida como "operador ternário") 87
Evite loops do/while 89
Retorno antecipado de uma função 91
O infame goto 91
Minimize o uso de aninhamentos 92

Você consegue acompanhar o fluxo de execução? 95
Sumário 96

Capítulo 8 ■ Divisão de expressões gigantes 97

Variáveis de explicação 98
Variáveis de resumo 98
Uso das leis de De Morgan 99
Uso excessivo da lógica de curto-circuito 100
Exemplo: Problemas com lógica complicada 101
Divisão de expressões gigantes 103
Outra forma criativa de simplificarmos expressões 105
Sumário 106

Capítulo 9 ■ Variáveis e legibilidade 107

Eliminação de variáveis 108
Reduza o escopo de suas variáveis 111
Prefira variáveis de gravação única 118
Um exemplo final 119
Sumário 121

Parte III ■ Reorganização de seu código 122

Capítulo 10 ■ Extração de subproblemas não relacionados 123

Exemplo introdutório: findClosestLocation() 124
Código utilitário puro 126
Outros códigos de propósito geral 127
Crie muitos códigos de propósito geral 129
Funcionalidades específicas de projetos 130
Simplificação de uma interface existente 131
Remodele uma interface de acordo com suas necessidades 132
Cuidado para não ir longe demais 133
Sumário 134

Capítulo 11 ■ Uma tarefa de cada vez 135

Tarefas podem ser pequenas 137
Extração de valores de um objeto 139
Um exemplo mais extenso 143
Sumário 146

Capítulo 12 ▪ Como transformar seus pensamentos em código.....147
Descreva sua lógica com clareza 148
Vale a pena conhecer suas bibliotecas 149
Aplicação desse método a problemas maiores..... 150
Sumário..... 154

Capítulo 13 ▪ Escreva menos código 156
Não se preocupe em implementar esse recurso – ele não será necessário..... 157
Questione e divida seus requisitos..... 157
Mantenha sua base de código pequena..... 159
Esteja familiarizado com as bibliotecas à disposição 161
Exemplo: Uso de ferramentas Unix em vez de codificação 163
Sumário..... 164

Parte V ▪ Tópicos selecionados.....165

Capítulo 14 ▪ Testes e legibilidade 166
Facilite a leitura e a manutenção de seus testes 167
O que há de errado com este teste?..... 167
Como tornar esse teste mais legível 168
Como tornar suas mensagens de erros mais legíveis..... 172
Escolha de boas entradas de teste.....174
Nomenclatura de funções de teste..... 176
O que havia de errado com aquele teste? 178
Desenvolvimento compatível com testes 179
Como perceber quando exageramos.....181
Sumário..... 182

Capítulo 15 ▪ Projeto e implementação de um "contador de minutos/horas" 183
O problema 184
Definição da interface da classe..... 184
Tentativa 1: uma solução ingênua..... 188
Tentativa 2: projeto da "esteira rolante"..... 191
Tentativa 3: um projeto com intervalos de tempo agrupados 194
Comparação das três soluções..... 199
Sumário.....200

Apêndice ■ Leituras adicionais 201
Livros que tratam da elaboração de códigos de alta qualidade 202
Livros que tratam de vários tópicos de programação 203
Livros de significado histórico 204
Sobre os autores 205

Índice remissivo 207