

Spis treści

<i>Przedmowa</i>	11
<i>Podziękowania</i>	12
<i>O książce</i>	13
<i>O autorze</i>	15

CZĘŚĆ I. SZEROKI HORYZONT 17

Rozdział 1. Cel testowania jednostkowego 19

1.1.	Obecna kondycja testowania jednostkowego	20
1.2.	Cel testowania jednostkowego	21
1.2.1.	Co czyni test dobrym?	23
1.3.	Stosowanie wskaźników pokrycia do mierzenia jakości zestawu testowego	24
1.3.1.	Interpretacja wskaźnika pokrycia kodu	25
1.3.2.	Interpretacja wskaźnika pokrycia gałęzi	26
1.3.3.	Problemy z pokryciem gałęzi	27
1.3.4.	Wymaganie procentowej wartości pokrycia	30
1.4.	Właściwości dobrego zestawu testowego	31
1.4.1.	Integracja z cyklem wytwarzania oprogramowania	31
1.4.2.	Koncentracja na najważniejszych częściach kodu	31
1.4.3.	Maksymalna wartość przy minimalnych kosztach	32
1.5.	Czego nauczysz się z tej książki	33
	Podsumowanie	34

Rozdział 2. Co to jest test jednostkowy? 37

2.1.	Definicja testu jednostkowego	38
2.1.1.	Izolacja — podejście londyńskie	38
2.1.2.	Izolacja — podejście klasyczne	44
2.2.	Klasyczna i londyńska szkoła testów jednostkowych	47
2.2.1.	Obsługa zależności według szkoły londyńskiej i klasycznej	47
2.3.	Zestawienie podejść — klasycznej i londyńskiej szkoły testowania jednostkowego	51
2.3.1.	Testowanie jednostkowe jednej klasy na raz	51
2.3.2.	Testowanie jednostkowe dużej mapy wzajemnie łączących się klas	52
2.3.3.	Dokładne wskazywanie źródła błędów	52
2.3.4.	Inne różnice między podejściem klasycznym a londyńskim	53

- 2.4. Testy integracyjne według dwóch szkół 54
 - 2.4.1. Testy systemowe to podzbiór testów integracyjnych 55
- Podsumowanie 57

Rozdział 3. Anatomia testu jednostkowego 59

- 3.1. Struktura testu jednostkowego 60
 - 3.1.1. Zastosowanie wzorca AAA 60
 - 3.1.2. Unikanie wielokrotnych sekcji przygotuj, zrób, sprawdź 61
 - 3.1.3. Unikanie warunków w testach 62
 - 3.1.4. Optymalna wielkość sekcji 63
 - 3.1.5. Liczba weryfikacji w sekcji asercji 65
 - 3.1.6. Sekwencja końcowa 65
 - 3.1.7. Zróźnicowanie systemu poddawanego testom 65
 - 3.1.8. Usunięcie komentarzy na temat sekcji z testów 66
- 3.2. Omówienie biblioteki testowej xUnit 67
- 3.3. Wielokrotne wykorzystanie jarzma testowego 68
 - 3.3.1. Silne wiązania między testami — antywzorzec 69
 - 3.3.2. Użycie konstruktora zmniejsza czytelność 70
 - 3.3.3. Lepszy sposób wielokrotnego wykorzystania jarzma testowego 70
- 3.4. Nazewnictwo testów jednostkowych 72
 - 3.4.1. Nazewnictwo testów jednostkowych — wytyczne 74
 - 3.4.2. Przykład: zmiana nazwy testu zgodnie z wytycznymi 74
- 3.5. Zamiana na testy parametryzowane 76
 - 3.5.1. Generowanie danych dla testów parametryzowanych 78
- 3.6. Biblioteka asercji i dalsze poprawianie czytelności testów 80
- Podsumowanie 81

CZĘŚĆ II. TESTY, KTÓRE PRACUJĄ DLA CIEBIE 83

Rozdział 4. Cztery filary dobrego testu jednostkowego 85

- 4.1. Cztery filary dobrego testu jednostkowego 86
 - 4.1.1. Filar pierwszy: ochrona przed regresją 86
 - 4.1.2. Filar drugi: odporność na zmiany 87
 - 4.1.3. Co powoduje wyniki obciążone błędem pierwszego rodzaju 89
 - 4.1.4. Skup się na końcowym wyniku, a nie szczegółach implementacyjnych 92
- 4.2. Nerozerwalny związek między pierwszą a drugą cechą 94
 - 4.2.1. Zwiększanie dokładności testów 94
 - 4.2.2. Waga wyników fałszywie dodatnich i fałszywie ujemnych — dynamika 96
- 4.3. Filary trzeci i czwarty: szybka informacja zwrotna i utrzymywalność 97
- 4.4. W poszukiwaniu idealnego testu 98
 - 4.4.1. Czy możliwe jest stworzenie idealnego testu 99
 - 4.4.2. Przypadek skrajny nr 1: test systemowy 99
 - 4.4.3. Przypadek skrajny nr 2: testy trywialne 100
 - 4.4.4. Przypadek skrajny nr 3: niestabilne testy 101
 - 4.4.5. W poszukiwaniu idealnego testu — wyniki 102

4.5.	Omówienie dobrze znanych pojęć z zakresu testów automatycznych	105
4.5.1.	Poziomy piramidy testów	105
4.5.2.	Wybór między testowaniem czarno- i białoskrzynkowym	107
	Podsumowanie	108

Rozdział 5. Atrapy i stabilność testów 111

5.1.	Rozróżnienie między atrapami a zaślepkami	112
5.1.1.	Rodzaje dublerów testowych	112
5.1.2.	Atrapa (narzędzie) kontra atrapa (dubler testowy)	113
5.1.3.	Nie poddawaj asercjom interakcji z zaślepkami	114
5.1.4.	Używanie atrap i zaślepek razem	116
5.1.5.	Związek atrap i zaślepek z poleceniami i zapytaniem	116
5.2.	Zachowanie dające się zaobserwować a szczegóły implementacyjne	117
5.2.1.	Dające się zaobserwować zachowanie to nie publiczny interfejs API	118
5.2.2.	Wyciekające szczegóły implementacyjne — przykład z operacją	119
5.2.3.	Dobrze zaprojektowany interfejs API i enkapsulacja	122
5.2.4.	Wyciekające szczegóły implementacyjne — przykład ze stanem	123
5.3.	Związek między atrapami a niestabilnością testów	125
5.3.1.	Architektura heksagonalna	125
5.3.2.	Komunikacja wewnątrzsystemowa i międzysystemowa	129
5.3.3.	Komunikacja wewnątrzsystemowa i międzysystemowa — przykład	130
5.4.	Klasyczna i londyńska szkoła testowania jednostkowego — raz jeszcze	133
5.4.1.	Nie wszystkie zewnętrzne zależności należy zastępować atrapami	133
5.4.2.	Wykorzystanie atrap do weryfikowania zachowania	135
	Podsumowanie	135

Rozdział 6. Style testowania jednostkowego 139

6.1.	Trzy style testowania jednostkowego	140
6.1.1.	Styl oparty na rezultatach — definicja	140
6.1.2.	Styl oparty na stanach — definicja	141
6.1.3.	Styl oparty na komunikacji — definicja	142
6.2.	Trzy style testowania jednostkowego — porównanie	143
6.2.1.	Porównanie stylów pod względem ochrony przed regresją i szybkości informacji zwrotnej	144
6.2.2.	Porównanie stylów pod względem odporności na zmiany	144
6.2.3.	Porównanie stylów pod względem utrzymywalności	145
6.2.4.	Porównanie stylów — wyniki	147
6.3.	Architektura funkcyjna	148
6.3.1.	Czym jest programowanie funkcyjne?	148
6.3.2.	Czym jest architektura funkcyjna?	151
6.3.3.	Porównanie architektury funkcyjnej i heksagonalnej	153
6.4.	Przejście do architektury funkcyjnej i testowania opartego na rezultatach	154
6.4.1.	System audytowania — wprowadzenie	154
6.4.2.	Wykorzystanie atrap w celu oddzielenia testu od systemu plików	157
6.4.3.	Przejście do architektury funkcyjnej	160
6.4.4.	Potencjalne dalsze kroki	164

- 6.5. Wady architektury funkcyjnej 165
 - 6.5.1. Zasadność stosowania architektury funkcyjnej 165
 - 6.5.2. Wady pod względem wydajności 167
 - 6.5.3. Wady pod względem rozmiaru bazy kodu 167
- Podsumowanie 168

Rozdział 7. Zmiany ku bardziej wartościowym testom jednostkowym 171

- 7.1. Określenie kodu podlegającego refaktoryzacji 172
 - 7.1.1. Cztery typy kodu 172
 - 7.1.2. Wykorzystanie wzorca Skromny Obiekt do podziału przeszacowanego kodu 175
- 7.2. Zmiany ku bardziej wartościowym testom 178
 - 7.2.1. System zarządzania kontaktami z klientami — wprowadzenie 178
 - 7.2.2. Próba nr 1: ujawnienie zależności 180
 - 7.2.3. Próba nr 2: wprowadzenie warstwy usług aplikacji 180
 - 7.2.4. Próba nr 3: usunięcie złożoności z usługi aplikacji 182
 - 7.2.5. Próba nr 4: wprowadzenie nowej klasy Company 184
- 7.3. Analiza optymalnego pokrycia testami jednostkowymi 186
 - 7.3.1. Testowanie warstwy domeny i kodu pomocniczego 187
 - 7.3.2. Testowanie kodu z pozostałych części diagramu 188
 - 7.3.3. Czy powinno się testować warunki wstępne? 188
- 7.4. Obsługa logiki warunkowej w kontrolerach 189
 - 7.4.1. Wykorzystanie wzorca Polecenie 191
 - 7.4.2. Wykorzystanie zdarzeń domeny do śledzenia zmian w modelu domeny 194
- 7.5. Wnioski 197
- Podsumowanie 199

CZĘŚĆ III. TESTY INTEGRACYJNE 203

Rozdział 8. Po co testy integracyjne? 205

- 8.1. Test integracyjny — definicja 206
 - 8.1.1. Rola testów integracyjnych 206
 - 8.1.2. Piramida testów — jeszcze raz 207
 - 8.1.3. Testy integracyjne kontra szybka reakcja 208
- 8.2. Które zewnętrzne zależności testować bezpośrednio 209
 - 8.2.1. Dwa typy zależności poza kontrolą procesu 210
 - 8.2.2. Obsługa zarządzanych i niez zarządzanych zależności 211
 - 8.2.3. Co, jeśli nie możesz wykorzystać prawdziwej bazy danych w testach integracyjnych? 212
- 8.3. Testy integracyjne — przykład 213
 - 8.3.1. Jakie scenariusze przetestować? 214
 - 8.3.2. Klasyfikacja bazy danych i szyny danych 214
 - 8.3.3. Co z testami systemowymi? 215
 - 8.3.4. Test integracyjny — próba pierwsza 216
- 8.4. Stosowanie interfejsów do abstrakcji zależności 217
 - 8.4.1. Interfejsy i luźne wiązania 217
 - 8.4.2. Po co używać interfejsów dla zewnętrznych zależności? 218
 - 8.4.3. Stosowanie interfejsów dla wewnętrznych zależności 219

8.5.	Najlepsze praktyki testów integracyjnych	220
8.5.1.	Jasno oznacz granice modelu domeny	220
8.5.2.	Zmniejszaj liczbę warstw	220
8.5.3.	Usuwać zapętłone zależności	222
8.5.4.	Użycie wielu sekcji działania w teście	224
8.6.	Jak testować zapisywanie logów	225
8.6.1.	Czy w ogóle powinno się testować pisanie logów	225
8.6.2.	Jak testować pisanie logów	226
8.6.3.	Ile logowania wystarczy	231
8.6.4.	Jak przekazywać instancje mechanizmu logowania	232
8.7.	Wnioski	233
	Podsumowanie	233

Rozdział 9. Najlepsze praktyki modelowania za pomocą atrap 237

9.1.	Maksymalizowanie wartości atrap	237
9.1.1.	Weryfikacja interakcji na obrzeżach systemu	240
9.1.2.	Zastępowanie atrap agentami	243
9.1.3.	Co z interfejsem IDomainLogger	245
9.2.	Najlepsze praktyki modelowania za pomocą atrap	246
9.2.1.	Atrapy służą tylko do testów integracyjnych	246
9.2.2.	Wiele atrap w jednym teście	246
9.2.3.	Weryfikacja liczby żądań	247
9.2.4.	Modeluj tylko typy, które sam utworzyłeś	247
	Podsumowanie	248

Rozdział 10. Testowanie bazy danych 251

10.1.	Warunki umożliwiające testowanie bazy danych	252
10.1.1.	Przechowywanie bazy danych w systemie kontroli wersji	252
10.1.2.	Dane referencyjne to część schematu bazy danych	253
10.1.3.	Oddzielne instancje dla każdego programisty	254
10.1.4.	Stanowe i migracyjne podejście do dostarczania bazy danych	254
10.2.	Zarządzanie transakcjami w bazie danych	256
10.2.1.	Zarządzanie transakcjami w kodzie produkcyjnym	256
10.2.2.	Zarządzanie transakcjami w testach integracyjnych	263
10.3.	Cykl życia danych testowych	265
10.3.1.	Równoległe i sekwencyjne wykonanie testów	265
10.3.2.	Sprzątanie danych pomiędzy wykonaniami testów	266
10.3.3.	Unikanie baz danych operujących w pamięci	267
10.4.	Wielokrotne wykorzystanie kodu w sekcjach	268
10.4.1.	Wielokrotne użycie kodu w sekcji przygotowań	268
10.4.2.	Wielokrotne użycie kodu w sekcji działania	271
10.4.3.	Wielokrotne użycie kodu w sekcji asercji	271
10.4.4.	Czy test tworzy zbyt wiele transakcji do bazy danych	272
10.5.	Często zadawane pytania na temat testowania baz danych	273
10.5.1.	Czy testować operacje odczytu?	273
10.5.2.	Czy testować repozytoria?	275
10.6.	Wnioski	276
	Podsumowanie	276

CZĘŚĆ IV. ANTYWZORCE TESTOWANIA JEDNOSTKOWEGO 279***Rozdział 11. Antywzorce testowania jednostkowego 281***

- 11.1. Testowanie jednostkowe prywatnych metod 282
 - 11.1.1. *Metody prywatne i niestabilność testów* 282
 - 11.1.2. *Metody prywatne i niedostateczne pokrycie* 282
 - 11.1.3. *Kiedy testowanie metod prywatnych jest akceptowalne* 283
- 11.2. Udostępnianie stanu prywatnego 285
- 11.3. Przenikanie wiedzy domenowej do testów 286
- 11.4. Zanieczyszczanie kodu 288
- 11.5. Modelowanie za pomocą atrap konkretnych klas 290
- 11.6. Praca z czasem 293
 - 11.6.1. *Czas jako kontekst środowiskowy* 293
 - 11.6.2. *Czas jako jawna zależność* 294
- 11.7. Wnioski 295
- Podsumowanie 295