

Spis treści (skrótowy)

Wprowadzenie	21
1. Witamy w krainie wzorców projektowych: <i>wprowadzenie</i>	33
2. Jak sprawić, by Twoje obiekty były zawsze dobrze poinformowane: <i>Wzorzec Obserwator</i>	67
3. Dekorowanie zachowania obiektów: <i>Wzorzec Dekorator</i>	109
4. Pizzeria zorientowana obiektowo: <i>Wzorzec Fabryka</i>	139
5. Obiekty jedyne w swoim rodzaju: <i>Wzorzec Singleton</i>	197
6. Hermetyzacja wywołań: <i>Wzorzec Polecenie</i>	217
7. Zdolność do adaptacji: <i>Wzorce Adapter oraz Fasada</i>	259
8. Hermetyzacja algorytmów: <i>Wzorzec Metoda Szablonowa</i>	297
9. Zarządzanie kolekcjami: <i>Wzorce Iterator i Kompozyt</i>	335
10. Stan obiektu: <i>Wzorzec Stan</i>	403
11. Kontrola dostępu do obiektu: <i>Wzorzec Proxy</i>	447
12. Łączenie wzorców: <i>Wzorce złożone</i>	517
13. Wzorce projektowe w praktyce: <i>Nowe życie z wzorcami</i>	595
14. Dodatek: <i>inne wzorce</i>	629
Skorowidz	649

Spis treści (z prawdziwego zdarzenia)



Wprowadzenie

Twój mózg jest skoncentrowany na wzorcach projektowych.

W tym rozdziale Ty starasz się czegoś dowiedzieć, a Twój mózg robi Ci przysługę i nie przykłada się do zapamiętywania zdobywanej wiedzy. Twój mózg myśli sobie: „Lepiej zostawię miejsce w pamięci na bardziej istotne informacje, na przykład: jakich dzikich zwierząt należy unikać bądź czy jeżdżenie nago na snowboardzie jest dobrym pomysłem”. A zatem w jaki sposób możesz przekonać swój mózg, że Twoje życie zależy od poznania wzorców projektowych?

Dla kogo przeznaczona jest ta książka?	22
Wiemy także, co sobie myśli Twój mózg	23
Metapoznanie	25
Zmusz swój mózg do posłuszeństwa	27
Zespół recenzentów technicznych	30
Podziękowania	31

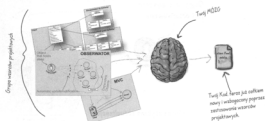
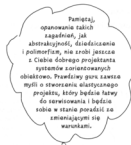
Wprowadzenie do wzorców projektowych

1

Witamy w krainie wzorców projektowych

Ktoś rozwiązał już Twoje problemy. W tym rozdziale dowiesz się, dlaczego (i w jaki sposób) możesz wykorzystać wiedzę i doświadczenia zdobyte przez innych projektantów i programistów, którzy podczas pracy nad różnymi projektami zmuszeni byli wstąpić na pełną zdradliwych pułapek ścieżkę i — co najważniejsze — udało im się przeżyć tę wyprawę. Zanim dobrniemy do końca rozdziału, rzucimy okiem na sposoby wykorzystywania wzorców projektowych i przedstawimy ich zalety, poznamy kilka podstawowych zasad projektowania zorientowanego obiektowo, a także omówimy sposób działania przykładowego wzorca. Najlepszą metodą zastosowania wzorca jest *załadowanie go bezpośrednio do Twojego mózgu*, a następnie zlokalizowanie obszarów w obrębie projektowanych rozwiązań oraz istniejących aplikacji, w których możesz go zastosować. Pracując z wzorcami projektowymi, zamiast wielokrotnego wykorzystywania tych samych fragmentów kodu wielokrotnie wykorzystujesz swoje doświadczenia.

Prosta aplikacja o nazwie SymulatorKaczki	34
Jacek rozmyśla o dziedziczeniu...	37
A może by tak interfejs?	38
Jedyny pewny element w procesie tworzenia oprogramowania	40
Oddzielanie tego, co się zmienia, od tego, co pozostaje niezmienione	42
Projektowanie zachowania Kaczki	43
Testowanie kodu klasy Kaczka	50
Dynamiczne ustawianie zachowania	52
Wielki diagram „ukrytych” zachowań	54
Relacja MA może być lepsza niż JEST	55
Rozmawiając o wzorcach projektowania	56
Potęga wspólnego słownika wzorców	60
W jaki sposób mogę wykorzystywać wzorce projektowe?	61
Twoja skrzynka narzędziowa	64
Rozwiązania ćwiczeń	66



Wzorzec Obserwator

2 Jak sprawić, by Twoje obiekty były zawsze dobrze poinformowane

Nie przepag okazji, kiedy dzieje się coś naprawdę ciekawego! Przedstawimy Ci wzorzec, który potrafi poinformować inne obiekty o tym, że wydarzyło się coś, czym powinny się zająć. Co ciekawe, obiekty mogą nawet samodzielnie decydować w czasie działania programu o tym, czy chcą być informowane o takich wydarzeniach. Wzorzec Obserwator jest jednym z najczęściej wykorzystywanych wzorców w pakiecie JDK (ang. *Java Development Kit*), a co najważniejsze — jest wręcz niewiarygodnie użyteczny. W niniejszym rozdziale rzucimy również okiem na relacje typu jeden-do-wielu oraz tzw. luźne związki (tak, to prawda, napisaliśmy „luźne związki”). Korzystając z wzorca Obserwator, z pewnością odmienisz swoje życie.

Podstawy programowania obiektowego

Abstrakcyjność

Hermetyzacja

Polimorfizm

Reguły programowania obiektowego

Poddawaj hermetyzacji to, co się zmienia.

Przedkładaaj kompozycję nad dziedziczenie.

Skoncentruj się na tworzeniu interfejsów, a nie implementacji.

Stawaj się twórcą projektów, w których obiekty są ze sobą luźno powiązane i o ile to możliwe, nie oddziałują na siebie wzajemnie.

Aplikacja sprawdzająca warunki pogodowe 69

Spotkanie z wzorcem Obserwator 74

Wydawca + Prenumeratorka = wzorzec Obserwator 75

Pięciominutowe przedstawienie — obserwowany kontra obserwujący 78

Definicja wzorca Obserwator 81

Siła luźnych zależności 83

Projektowanie stacji meteorologicznej 86

Implementacja stacji meteorologicznej 87

Java — zastosowanie wbudowanego wzorca Obserwator 94

Ciemna strona klasy `java.util.Observable` 101

Twoja skrzynka narzędziowa 104

Rozwiązania ćwiczeń 107

Obiekt, którego stan
jest obserwowany

RELACJA JEDEN-DO-WIELU

Automatyczna aktualizacja
(powiadomienie)

Obiekty zależne

Wzorzec Dekorator

3

Dekorowanie zachowania obiektów

W zasadzie niniejszy rozdział możemy równie dobrze zatytułować „Otwieranie oczu programistom z nadmiernymi skłonnościami do używania dziedziczenia”.

W tym rozdziale spróbujemy krytycznie przyrzeć się zwyczajowym skłonnościom do nadużywania mechanizmu dziedziczenia oraz nauczymy Cię sposobów dekorowania zachowania klas w czasie działania programu przy użyciu pewnej formy kompozycji obiektów. Dlaczego? Po zapoznaniu się z technikami dekoracji zachowania klas będziesz mógł wyposażać swoje (i nie tylko) obiekty w nowe możliwości bez konieczności dokonywania jakichkolwiek modyfikacji w kodzie klas podstawowych.

Witamy w „Star Café”	110
Reguła otwarte-zamknięte	116
Spotkanie z wzorcem Dekorator	118
Konstruowanie zamówienia przy użyciu Dekoratorów	119
Definicja wzorca Dekorator	121
Dekorujemy nasze Napoje	122
Tworzymy kod aplikacji „Star Café”	125
Dekoratory w świecie rzeczywistym: obsługa wejścia-wyjścia w języku Java	130
Tworzenie własnych dekoratorów obsługi wejścia-wyjścia	132
Tvoja skrzynka narzędziowa	135
Rozwiązania ćwiczeń	136

Zawsze sądziłem,
że prawdziwi mężczyźni tworzą
podkłasy dla wszystkiego, co się tylko
do tego nadaje. Tak było do czasu,
gdy dowiedziałem się o korzyściach,
jakie daje rozszerzanie możliwości
aplikacji na poziomie działania,
a nie kompilacji. A teraz
— spojrzcie tylko na mnie!

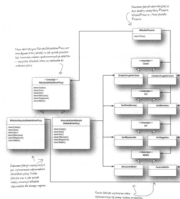


Wzorzec Fabryka

4

Pizzeria zorientowana obiektowo

Przygotuj się do stworzenia kilku projektów, w których zastosujemy różne powiązania pomiędzy poszczególnymi obiektami. Stworzenie nowego obiektu to dużo więcej niż tylko proste zastosowanie operatora `new`. Niebawem przekonasz się, że proces ten jest operacją, która nie zawsze powinna być publicznie dostępna, a co więcej, jest operacją, która często może prowadzić do poważnych problemów z powiązaniem międzyobiekto- wymi. A tego byś nie chciał, prawda? Przekonaj się, w jaki sposób wzorzec Factory może uratować Cię z takiej opresji.



Kiedy widzisz „nowy” obiekt, myśl o nim jako o „konkretnym”	140
Pizza w Obiektywie	142
Hermetyzacja procesu tworzenia obiektów	144
Budujemy prostą fabrykę pizzy	145
Tworzymy definicję „wzorca” Simple Factory	147
Nowa struktura Pizzerii	150
Zezwalamy klasom podrzędnym na podejmowanie decyzji	151
Tworzymy Pizzerię	153
Deklarowanie metody typu Fabryka	155
Spotkanie z wzorcem Metoda Fabrykująca	161
Równoległa hierarchia klas	162
Definicja wzorca Metoda Fabrykująca	164
Pizzeria mocno uzależniona	167
Sprawdzamy zależności pomiędzy obiektami	168
Zastosowanie reguły DIP	170
A w międzyczasie, na zapleczu Pizzerii...	174
Rodziny składników...	175
Budujemy fabryki składników pizzy	176
Fabryka Abstrakcyjna	183
Za kulisami	184
Definicja wzorca Fabryka Abstrakcyjna	186
Porównanie Metody Fabrykującej oraz Fabryki Abstrakcyjnej	190
Twoja skrzynka narzędziowa	192
Rozwiązania ćwiczeń	193

Wzorzec Singleton

5

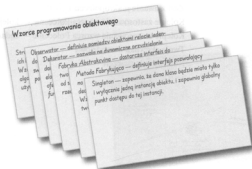
Obiekty jedyne w swoim rodzaju

Kolejnym przystankiem w naszej podróży jest wzorzec Singleton, czyli nasza przepustka do kreowania jedynych w swoim rodzaju obiektów, posiadających tylko jedną instancję. Być może ucieszysz się na wieść o tym, że Singleton jest najprostszym z istniejących wzorców projektowych (przynajmniej pod względem kategorii stopnia złożoności jego diagramu klas); jak by na to nie patrzeć, jego diagram składa się tylko z jednej klasy! Ale nie wpadaj w euforię; niezależnie od prostoty diagramu klas tego wzorca na drodze prowadzącej do jego implementacji napotkamy całkiem sporo wybojów i dziur. Lepiej zapnij mocno pasy — to nie będzie takie proste, jak mogłoby się wydawać.



Hershey, PA

Jeden i tylko jeden	198
Mały Singleton	199
Analiza klasycznej implementacji wzorca Singleton	201
Wyznania obiektu Singleton	202
Fabryka czekolady	203
Definicja wzorca Singleton	205
Ups, mamy problem...	206
Zostań wirtualną maszyną Java	207
Jak sobie radzić z wielowątkowością?	208
Wzorzec Singleton — pytania i odpowiedzi	212
Twoja skrzynka narzędziowa	214
Rozwiązania ćwiczeń	216



Wzorzec Polecenie

6

Hermetyzacja wywołań

W niniejszym rozdziale przeniesiemy hermetyzację na zupełnie nowy poziom: mamy zamiar dokonać hermetyzacji wywołań metod. Zgadza się, dzięki hermetyzacji wywołań metod możemy wykrystalizować pewne fragmenty obliczeń tak, że obiekt wywołujący obliczenia nie musi się martwić, w jaki sposób je wykonać; po prostu wykorzystuje naszą metodę. Z takimi hermetyzowanymi wywołaniami metod możemy również dokonywać wielu zadziwiająco sprytnych operacji, takich jak na przykład zapisywanie ich do dzienników czy też ponowne wykorzystywanie w celu zaimplementowania mechanizmu Cofnij (ang. Undo) w naszej aplikacji.



Automatyka w domu i zagrodzie	218
Mamy nową zabawkę! Sprawdzamy, jak działa SuperPilot...	219
Co zawiera otrzymany dysk CD-R?	220
A w międzyczasie w naszym barze szybkiej obsługi...	223
Przyjrzyjmy się nieco dokładniej wzajemnym interakcjom...	224
Zadania i zakresy odpowiedzialności	225
Od Baru do wzorca Polecenie	227
Nasze pierwsze POLECENIE	229
Definicja wzorca Polecenie	232
Wzorzec Command i SuperPilot	234
Implementujemy SuperPilota	236
Sprawdzamy możliwości naszego SuperPilota	238
Nadszedł wreszcie czas, aby utworzyć trochę dokumentacji...	241
Implementacja mechanizmu wycofywania przy użyciu stanów	246
Każdy pilot powinien posiadać tryb Impreza!	250
Zastosowanie makropoleceń	251
Kolejne zastosowania wzorca Polecenie — kolejkovanie żądań	254
Kolejne zastosowania wzorca Polecenie — żądania rejestracji	255
Twoja skrzynka narzędziowa	256
Rozwiązania ćwiczeń	257

Wzorce Adapter oraz Fasada

7

Zdolność do adaptacji

W niniejszym rozdziale mamy zamiar dokonać paru niesamowitych wyczynów z dziedziny rzeczy niemożliwych, takich jak na przykład włożenie kwadratowego kołka do okrągłego otworu. Brzmi nierealnie? Nie wtedy, kiedy mamy pod ręką odpowiednie wzorce projektowe. Pamiętajsz wzorec Dekorator? Podczas pracy z nim owijaliśmy obiekty innymi obiektami tak, aby nadać im nowe zachowania. Teraz mamy zamiar postępować tak samo, ale w nieco innym celu: chcemy sprawić, by ich interfejsy wyglądały jak coś, czym nie są. Dlaczego jednak mielibyśmy to robić? Na przykład po to, aby zaadaptować projekt oczekujący danego interfejsu do klasy, która implementuje zupełnie inny interfejs. To jeszcze nie wszystko; skoro już jesteśmy przy tym temacie, przyjrzymy się również innemu wzorcowi, który owija obiekty w celu uproszczenia ich interfejsów.

Europejski standard ściennego gniazda elektrycznego



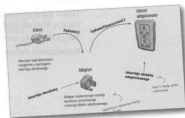
Adapter



Standardowa wtyczka zasilająca



Adaptory są wśród nas	260
Adaptory zorientowane obiektowo	261
Wzorec Adapter bez tajemnic	265
Definicja wzorca Adapter	267
Adaptory obiektów i klas	268
Temat dzisiejszej wieczornej pogawędki: Adapter obiektów i Adapter klas	271
Adaptory w świecie rzeczywistym	272
Adaptujemy interfejs Enumeration do wymagań interfejsu Iterator	273
Temat dzisiejszej wieczornej pogawędki: wzorce Dekorator i Adapter	276
Nie ma to jak kino domowe	279
Światła, kamera, fasada!	282
Konstruujemy fasadę naszego systemu kina domowego	285
Definicja wzorca Fasada	288
Reguła ograniczania interakcji	289
Twoja skrzynka narzędziowa	294
Rozwiązania ćwiczeń	296



8

Hermetyzacja algorytmów

Jesteśmy jak w transie: hermetyzowaliśmy już proces tworzenia obiektów, wywołania metod, złożone interfejsy, kaczki, indyki, pizze... ciekawe, co będzie następne? Otóż teraz mamy zamiar zająć się hermetyzacją fragmentów algorytmów, tak aby klasy podrzędne mogły „podczepiać się” w różnych miejscach wykonywanych obliczeń. Co więcej, zajmiemy się również regułą projektowania, której korzenie wywodzą się w prostej linii z... Hollywood.

Tworzymy klasy reprezentujące kawę i herbatę (w języku Java)

Kawa i herbata, czyli klasy abstrakcyjne

Ciagniemy nasz projekt o krok dalej...

Wydobywanie metody recepturaParzenia()

Czego już dokonaliśmy?

Spotkanie z wzorcem Metoda Szablono

Zróbmy sobie herbatę...

Co nam daje zastosowanie metody szablonowej?

Definicja wzorca Metoda Szablono

Bliskie spotkania z kodem aplikacji

Haczyk na wzorec Metoda Szablono...

Zastosowanie haczyka

Testujemy naszą aplikację

Reguła Hollywood

Reguła Hollywood a wzorec Metoda Szablono

Wzorec Metoda Szablono w głębokiej kniei...

Sortowanie przy użyciu wzorca Metoda Szablono

A teraz musimy posortować trochę kaczek...

Porównywanie kaczek z innymi kaczkami

Robimy maszynę do sortowania kaczek

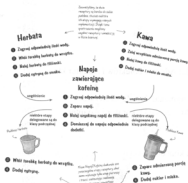
Zabawy z ramkami

Aplety Java

Temat dzisiejszej wieczornej pogawędki: wzorec Metoda Szablono oraz Strategia

Twoja skrzynka narzędziowa

Rozwiązania ćwiczeń



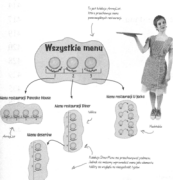
Wzorce Iterator i Kompozyt

9

Zarządzanie kolekcjami

Jest wiele sposobów grupowania obiektów w kolekcje. Można utworzyć obiekty `Array`, `Stack`, `List`, `Hashtable`. Każdy z nich ma swoje zalety i wady. Jednak w pewnym momencie klient rozpocznie iteracyjne przetwarzanie elementów kolekcji. Czy wtedy ujawnisz mu swoją implementację? Mam nadzieję, że nie. To nie byłoby profesjonalne. Nie musisz się jednak obawiać, Twoja kariera zawodowa nie jest zagrożona. W tym rozdziale przedstawimy metodę, która umożliwia klientom przetwarzanie iteracyjne bez wiedzy o tym, jak obiekty są przechowywane. Przedstawimy też technikę tworzenia *superkolekcji* (ang. *super collections*) obiektów, które pozwalają na obsługę bardzo rozbudowanych struktur danych. Będziemy także pisać o odpowiedzialności obiektów.

Fuzja restauracji Diner i Pancake House	336
Implementacje menu Łukasza i Miłosza	338
Czy można hermetyzować iterację?	343
Wzorzec Iterator	345
Wiązanie iteratora z obiektem menu	347
Co już mamy... Szersze spojrzenie na kod naszego projektu	351
Uproszczenia po wprowadzeniu interfejsu <code>java.util.Iterator</code>	353
Jaki jest efekt końcowy?	355
Definicja wzorca Iterator	356
Jeden zakres odpowiedzialności	359
Iteratory i kolekcje	368
Iteratory i kolekcje w języku Java 5	369
I gdy już miało być tak dobrze...	373
Definicja wzorca Kompozyt	376
Projektujemy menu oparte na wzorcu Kompozyt	379
Implementacja klasy Menu	382
Powracamy do iteratora	388
IteratorPusty	392
Wzorce Iterator i Kompozyt razem...	394
Twoja skrzynka narzędziowa	399
Rozwiązania ćwiczeń	400



Wzorec Stan

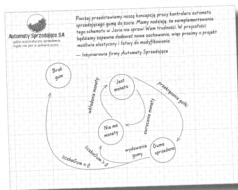
10

Stan obiektu

Mало znany fakt: wzorce Strategy i State to bliźniaki, rozdzielone zaraz po narodzinach. Jak już wiemy, wzorec Strategy umożliwił przeprowadzenie wielu niezwykle udanych przedsięwzięć opartych na zamiennie stosowanych algorytmach. Wzorec State ma inną rolę. Jest nią wspomaganie obiektów w kontrolowaniu ich własnych zachowań poprzez wewnętrzną zmianę stanu. Łatwo usłyszeć, jak mówi swoim podopiecznym: „Powtarzaj za mną: jestem wystarczająco zdolny, jestem wystarczająco dobry, dam radę to zrobić...”.



Krótką narada	405
Maszyny stanowe 101	406
Piszemy kod	408
Wiedziałeś, że to jest blisko... zmiana!	412
Kłopotliwy STAN rzeczy...	414
Definiowanie interfejsów i klas reprezentacji stanu	417
Implementowanie klas Stan	419
Nowa wersja automatu sprzedającego	420
Definicja wzorca Stan	428
Wzorec Stan kontra wzorec Strategy	429
Wzorec Stan, weryfikacja projektu	435
Niemal zapomnieliśmy!	438
Twoja skrzynka narzędziowa	441
Rozwiązania ćwiczeń	442



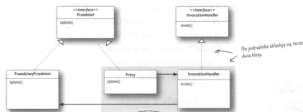
11

Kontrola dostępu do obiektu

Próbowałeś kiedyś stosować metodę „dobrego i złego”? Ty jesteś tym dobrym, który zrobi wszystko, o co się go poprosi, który jest zawsze miły i uprzejmy. Nie chcesz jednak, żeby każdy mógł prosić o Twoje usługi. To jest miejsce dla „złego”, który będzie kontrolował dostęp do Ciebie. Takie jest właśnie zadanie pośredników (ang. *proxy*) w modelu obiektowym — kontrolowanie dostępu i zarządzanie nim. Jak się przekonamy, istnieje *bardzo* wiele schematów takiego pośrednictwa. Obiekty Proxy mogą przekazywać wywołanie metody obiektowi w innym węzle internetu; bywa też, że zastępują wyjątkowo leniwe obiekty.



Kontrolowanie stanu automatów sprzedających	448
Rola „zdalnego pośrednika”	452
RMI — wycieczka z przewodnikiem	455
Zdalny pośrednik automatu sprzedającego	468
Pośrednik zdalny, za kulisami	476
Definicja wzorca Proxy	478
Pośrednik wirtualny	480
Projektowanie wirtualnego pośrednika do wyświetlania okładek	482
Pośrednik wirtualny, za kulisami	488
Wykorzystanie mechanizmów Java API	492
Teatrzyk — ochrona przedmiotów	496
Budowanie dynamicznego pośrednika	497
ZOO pośredników	506
Twoja skrzynka narzędziowa	508
Rozwiązania ćwiczeń	509



12

Łączenie wzorców

Przyszłoby Ci do głowy, że wzorce mogą pracować razem? Byliśmy już świadkami wielu niespokojnych „Pogawędek przy kominku” (a ominął Cię „Death Match” wzorców, który wydawca kazał wyrzucić) — czy wsłuchując się w ich ton, można jeszcze liczyć na to, że wzorce będą ze sobą współpracować? Możesz wierzyć lub nie, ale najbardziej wyszukane projekty obiektowe wykorzystują wiele wzorców jednocześnie. Przygotuj się na kolejny poziom wiedzy o wzorcach projektowych. Czas na wzorce złożone.

Wzorec złożony

518

Powrót kaczek

519

Potrzebujemy adaptera gęsi

522

Wprowadzamy zliczanie kwaknięć

524

Fabryka produkująca kaczkę

526

Tworzymy stado kaczek

531

Przygotowanie interfejsu Observable

534

Co zrobiliśmy?

541

Widok z lotu kaczki — diagram klas

542

Model-Widok-Kontroler — piosenka

544

Kluczem do schematu MVC będą wzorce projektowe

546

Spojrzenie na schemat Model-Widok-Kontroler przez pryzmat wzorców

550

Wykorzystujemy MVC do sterowania beatem...

552

Piszemy kod elementów

555

Widok

557

A teraz kontroler

560

Eksplorujemy możliwości wzorca Strategia

563

Adaptowanie modelu

564

Nowy kontroler — SerceKontroler

565

Wzorec MVC i sieć WWW

567

Model 2 a wzorce projektowe

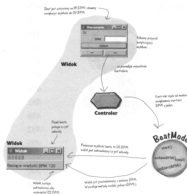
575

Twoja skrzynka narzędziowa

578

Rozwiązania ćwiczeń

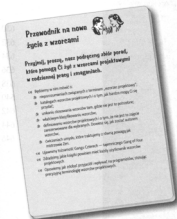
579



Nowe życie z wzorcami

13 Wzorce projektowe w praktyce

Ach, jesteś już gotowy na spotkanie z nowym wspaniałym światem pełnym wzorców projektowych... Ale zanim rozpoczniesz wędrówkę ku nowym horyzontom, poświęć chwilę na przeczytanie rozdziału poświęconego pewnym szczególnym kwestiom, które pojawiają się, gdy rozpoczynasz stosowanie wzorców w codziennej pracy. Nie wszędzie jest tak pięknie, jak w Obiektowie. Przygotowaliśmy więc mały przewodnik, który pomoże Ci odnaleźć się w twardej rzeczywistości...



Przewodnik na nowe życie z wzorcami	596
Definicja wzorca projektowego	597
Drugie spojrzenie na definicję wzorca	599
Niech moc będzie z Tobą	600
Katalog wzorców	601
Jak tworzyć wzorce	604
Zostać autorem wzorców projektowych	605
Porządkowanie wzorców projektowych	607
Myślenie wzorcami	612
Głowa pełna wzorców	615
Nie zapominaj o potężde jednolitego słownictwa	617
Pięć podstawowych sposobów promowania Twojego słownictwa	618
Gang Czterech w Obiektowie	619
Podróż dopiero się zaczyna...	620
Inne źródła informacji o wzorcach	621
ZOO pełne wzorców	622
Walka ze złem przy użyciu antywzorców	624
Tvoja skrzynka narzędziowa	626
Opuszczamy Obiektowo...	627



