Spis treści

Przedmowa 9

Podziękowania 11

O książce 13

O autorze 16

CZĘŚĆ I. DLACZEGO TO WSZYSTKO MA ZNACZENIE 17

Rozdział 1. Szersze spojrzenie 19

1.1. Python jest językiem dla przedsiębiorstw 20

1.1.1. Czasy się zmieniają 20

1.1.2. Co lubię w Pythonie 21

1.2. Python jest językiem przyjaznym do nauczania 21

1.3. Projektowanie jest procesem 22

1.3.1. Doświadczenie użytkownika 23

1.3.2. Już to widziałeś 24

1.4. Projektowanie umożliwia tworzenie lepszego oprogramowania 24

1.4.1. Rozważania przy projektowaniu oprogramowania 25

1.4.2. Oprogramowanie organiczne 26

1.5. Kiedy inwestować w projektowanie 27

1.6. Nowe początki 28

1.7. Projekt jest demokratyczny 29

1.7.1. Obecność umysłu 29

1.8. Jak korzystać z tej książki 31

Podsumowanie 32

CZĘŚĆ II. PODSTAWY PROJEKTOWANIA 33

Rozdział 2. Rozdzielanie zagadnień 35

2.1. Przestrzenie nazw 36

2.1.1. Przestrzenie nazw oraz polecenie import 36

2.1.2. Wiele twarzy importowania 38

2.1.3. Przestrzenie nazw zapobiegają kolizjom nazw 39

2.2. Hierarchia rozdzielania w Pythonie 41

2.2.1. Funkcje 41

2.2.2. Klasy 47

2.2.3. Moduły 52

2.2.4. Pakiety 52

Podsumowanie 54

Rozdział 3. Abstrakcja i hermetyzacja 57

3.1. Co to jest abstrakcja? 57

3.1.1. "Czarna skrzynka" 57

3.1.2. Abstrakcja jest jak cebula 59

3.1.3. Abstrakcja to uproszczenie 61

3.1.4. Dekompozycja umożliwia zastosowanie abstrakcji 62

3.2. Hermetyzacja 63

3.2.1. Konstrukty hermetyzacji w Pythonie 63

3.2.2. Prywatność w Pythonie 64

3.3. Wypróbuj 64

3.3.1. Refaktoryzacja 66

3.4. Style programowania to też abstrakcja 67

3.4.1. Programowanie proceduralne 67

3.4.2. Programowanie funkcyjne 67

3.4.3. Programowanie deklaratywne 69

3.5. Typowanie, dziedziczenie i polimorfizm 70

3.6. Rozpoznanie nieprawidłowej abstrakcji 72

3.6.1. Kwadratowe kołki i okrągłe otwory 72

3.6.2. Buty szyte na miarę 73

Podsumowanie 73

Rozdział 4. Projektowanie pod kątem wysokiej wydajności 75

4.1. Pędząc przez czas i przestrzeń 76

4.1.1. Złożoność jest trochę... złożona 76

4.1.2. Złożoność czasowa 77

4.1.3. Złożoność przestrzeni 80

4.2. Wydajność i typy danych 81

4.2.1. Typy danych dla stałego czasu 81

4.2.2. Typy danych w czasie liniowym 82

4.2.3. Złożoność przestrzeni w operacjach na typach danych 82

4.3. Zrób to, zrób to dobrze, spraw, żeby było szybkie 85

4.3.1. Zrób to 86

4.3.2. Zrób to dobrze 86

4.3.3. Spraw, żeby było szybkie 89

4.4. Narzędzia 89

4.4.1. timeit 90

4.4.2. Profilowanie CPU 91

4.5. Wypróbuj 92

Podsumowanie 93

Rozdział 5. Testowanie oprogramowania 95

5.1. Czym jest testowanie oprogramowania 96

5.1.1. Czy robi to, co napisano w instrukcji 96

5.1.2. Anatomia testu funkcjonalnego 96

5.2. Podejścia do testowania funkcjonalnego 98

5.2.1. Testy manualne 98

5.2.2. Testy automatyczne 98

5.2.3. Testy akceptacyjne 99

5.2.4. Testy jednostkowe 100

5.2.5. Testy integracyjne 101

5.2.6. Piramida testów 102

5.2.7. Testy regresji 103
5.3. Stwierdzenie faktów 104
5.4. Testy jednostkowe z unittest 105
5.4.1. Organizacja testów z unittest 105
5.4.2. Uruchamianie testów z unittest 105
5.4.3. Pisanie pierwszego testu w unittest 105
5.4.4. Pierwszy test integracyjny w unittest 108
5.4.5. Zamienniki testowe 110
5.4.6. Wypróbuj 112
5.4.7. Pisanie ciekawych testów 114
5.5. Testowanie z pytest 114
5.5.1. Organizowanie testów w pytest 115
5.5.2. Konwersja testów w unittest na pytest 115
5.6. Poza testowaniem funkcjonalnym 116
5.6.1. Testy wydajności 116
5.6.2. Testowanie obciążenia 117
5.7. Rozwój oparty na testach: podstawy 117
5.7.1. To sposób myślenia 118
5.7.2. To filozofia 118
Podsumowanie 119

CZĘŚĆ III. ARANŻACJA DUŻYCH SYSTEMÓW 121

Rozdział 6. Rozdzielanie aspektów w praktyce 123
6.1. Aplikacja do tworzenia zakładek z wiersza poleceń 124
6.2. Wycieczka po Bark 125
6.2.1. Korzyści wynikające z rozdzielenia: powtórzenie 125
6.3. Początkowa struktura kodu, według aspektów 126
6.3.1. Warstwa przechowywania danych 127
6.3.2. Warstwa logiki biznesowej 136
6.3.3. Warstwa prezentacji 140

Podsumowanie 147

Rozdział 7. Rozszerzalność i elastyczność 149

7.1. Co to jest kod rozszerzalny? 149

7.1.1. Dodawanie nowych zachowań 150

7.1.2. Modyfikacja istniejących zachowań 152

7.1.3. Luźne wiązanie 153

7.2. Rozwiązania dla sztywności 155

7.2.1. Oddawanie: odwrócenie sterowania 155

7.2.2. Diabeł tkwi w szczegółach: poleganie na interfejsach 158

7.2.3. Zwalczanie entropii: zasada odporności 159

7.3. Ćwiczenie rozszerzalności 160

Podsumowanie 164

Rozdział 8. Zasady (i wyjątki) dziedziczenia 165

8.1. Historia dziedziczenia w programowaniu 165

8.1.1. Panaceum 166

8.1.2. Wyzwania hierarchii 166

8.2. Dziedziczenie obecnie 168

8.2.1. Do czego służy dziedziczenie 168

8.2.2. Zastępowalność 169

8.2.3. Idealny przypadek użycia dziedziczenia 170

8.3. Dziedziczenie w Pythonie 173

8.3.1. Inspekcja typu 173

8.3.2. Dostęp do klasy bazowej 174

8.3.3. Wielokrotne dziedziczenie i kolejność rozwiązywania metod 174

8.3.4. Abstrakcyjne klasy bazowe 178

8.4. Dziedziczenie i kompozycja w programie Bark 180

8.4.1. Refaktoryzacja w celu użycia abstrakcyjnej klasy bazowej 180

8.4.2. Ostateczne spojrzenie na wykonane dziedziczenie 182

Podsumowanie 182

Rozdział 9. Zapewnianie lekkości 183
9.1. Jak duża powinna być klasa/funkcja/moduł 183
9.1.1. Fizyczny rozmiar 184
9.1.2. Pojedyncza odpowiedzialność 184
9.1.3. Złożoność kodu 185
9.2. Rozkładanie złożoności 189
9.2.1. Wyodrębnianie konfiguracji 189
9.2.2. Wyodrębnianie funkcji 191
9.3. Dekompozycja klas 193
9.3.1. Złożoność inicjacji 193
9.3.2. Wyodrębnianie klas i przekazywanie wywołań 195
Podsumowanie 199

Rozdział 10. Luźne wiązania w praktyce 201
10.1. Definicja wiązania 201
10.1.1. Tkanka łączna 201
10.1.2. Ścisłe wiązania 202
10.1.3. Luźne wiązania 205
10.2. Rozpoznawanie wiązania 208
10.2.1. Zazdrość o funkcje 208
10.2.2. Chirurgia przy użyciu strzelby 209
10.2.3. Nieszczelne abstrakcje 209
10.3. Wiązania w programie Bark 210
10.4. Radzenie sobie z wiązaniami 212
10.4.1. Powiadamianie użytkownika 212
10.4.2. Przechowywanie zakładek 215
10.4.3. Wypróbuj 216
Podsumowanie 219

CZĘŚĆ IV. CO DALEJ? 221

Rozdział 11. Dalej i wyżej 223

11.1. Co teraz 223

11.1.1. Opracuj plan 224

11.1.2. Wykonaj plan 225

11.1.3. Śledź swoje postępy 227

11.2. Wzorce projektowe 228

11.2.1. Mocne i słabe strony wzorców projektowych w Pythonie 229

11.2.2. Tematy, od których zacząć 230

11.3. Systemy rozproszone 230

11.3.1. Tryby awarii w systemach rozproszonych 231

11.3.2. Adresowanie stanu aplikacji 232

11.3.3. Tematy, od których zacząć 232

11.4. Zanurz się głęboko w Pythonie 232

11.4.1. Styl kodu w języku Python 232

11.4.2. Funkcje językowe są wzorcami 233

11.4.3. Tematy, od których zacząć 234

11.5. Gdzie byłeś 234

11.5.1. Tam i z powrotem: opowieść programisty 234

11.5.2. Zakończenie 236

Podsumowanie 236

Załącznik A. Instalacja Pythona 237

A.1. Jakiej wersji Pythona powinienem użyć 237

A.2. Python "systemowy" 238

A.3. Instalowanie innych wersji Pythona 238

A.3.1. Pobierz oficjalną dystrybucję Pythona 238

A.3.2. Pobierz za pomocą Anacondy 239

A.4. Weryfikacja instalacji 240